Freeform Search

Database:
US Pre-Grant Publication Full-Text Database
US Patents Full-Text Database
US OCR Full-Text Database
EPO Abstracts Database
JPO Abstracts Database
Derwent World Patents Index
IBM Technical Disclosure Bulletins

Term: L7 and (assign$ with IP with address$ with (virtual adj1 network$))

Display: 10 **Documents in Display Format:** KWIC **Starting with Number** 1

Generate: ○ Hit List ◉ Hit Count ○ Side by Side ○ Image

Search Clear Interrupt

Search History

DATE: Tuesday, May 17, 2005 Printable Copy Create Case

Set Name side by side	Query	Hit Count	Set Name result set
	DB=USPT; PLUR=YES; OP=ADJ		
L10	L7 and (assign$ with IP with address$ with (virtual adj1 network$))	2	L10
L9	L7 and (assign$ with IP with address$ with virtual with network$)	10	L9
L8	L7 and (assign$ with IP with address$)	157	L8
L7	L1 and (virtual adj2 network$)	826	L7
L6	L4 and (virtual with network$)	0	L6
L5	L4 and (assign$ with IP with address$)	1	L5
L4	5978568.pn.	1	L4
L3	L2 and (isolat$ or separat$)	1	L3
L2	L1 and (assign$ IP with address$ with virtual with network$)	8	L2
L1	709/$.ccls.	17870	L1

END OF SEARCH HISTORY

L10: Entry 2 of 2 File: USPT Mar 31, 1998

DOCUMENT-IDENTIFIER: US 5734865 A
TITLE: Virtual local area network well-known port routing mechanism for mult--emulators in an open system environment

Abstract Text (1):
A local host data processing system operating under the control of a local host operating system includes components of multiple emulating hosted operating systems. The host operating system further include a TCP/IP network protocol stack which couples to the communications facilities of the host system connected to a local area network for communicating with a number of remote host systems. Host and hosted operating systems share the same TCP/IP network protocol stack. A virtual network mechanism is configured within the local host system to be operatively coupled to the host network protocol stack and provide access to well-known port application programs. When so configured, the mechanism functions as another LAN to which multiple virtual host systems are attached for executing applications under control of the emulating hosted operating systems. The mechanism transforms the well-known port identifier of each inbound packet into a non-well-known port identifier in addition to other station address identifier fields. It then redirects the transformed packet back to the IP layer of the stack for transfer to the appropriate well-known port application program being run by the hosted operating system of the particular virtual host system. The mechanism reverses this operation for each reply packet which it redirects back to the IP layer for forwarding to the remote system. This eliminates the need to specify additional protocol stacks and to provide additional communication hardware facilities for handling multiple instances of well-known port applications programs running on the different virtual host/multiple hosted operating systems.

Parent Case Text (1):
This is a continuation-in-part of patent application Ser. No. 08/473,476, filed on Jun. 7, 1995, now U.S. Pat. No. 5,636,371, issued on Jun. 3, 1997, entitled, "Virtual Network Mechanism to Access Well Known Port Application Programs Running on a Single Host System", invented by Kin C. Yu, U.S. Pat. No. 5,636,371.

Brief Summary Text (18):
The above and other objects of the present invention are achieved in a preferred embodiment of the virtual network mechanism of the present invention which operates under the control of a host operating system, as for example, an enhanced version of the UNIX operating system running on a local host computer system which connects to a local area network (LAN) or internetwork for communicating with a number of remote host systems using a standard communications protocol In the preferred embodiment, the host system also includes the components of a plurality of hosted operating system components, such as for example, multiple instances of an emulator.

Brief Summary Text (19):
The host operating system further includes a communications network protocol stack which in the preferred embodiment corresponds to a host TCP/IP protocol stack. Both each hosted and host application programs share the single protocol stack. The virtual network mechanism of the present invention resolves the naming conflict

arising from the use of multiple instances of well-known port application programs being run by each hosted and host operating systems.

Brief Summary Text (21):
The mechanism of the present invention is configured within the host operating system as a separate network interface which couples to the network protocol stack just as "another physical network." This allows the mechanism to make use of the standard internetwork gateway functionality associated with such communication networks. The IP layer routes each packet addressed to the specific hosted system to the virtual network mechanism as if it were another network (i.e., as if the packets were being transferred from one network to another network through an internetwork gateway).

Brief Summary Text (22):
More specifically, the virtual network mechanism utilizes a different set of control data structures corresponding to a different one of the virtual host systems. Each set of structures includes an interface network structure used for connecting the virtual network mechanism to the network protocol stack, a control structure which represents the particular virtual host system and a client table structure which is used to process client requests directed to the virtual host system by a remotely located client system. By configuring these different sets of structures to operate as a corresponding number of virtual host systems, this enables the routines of the virtual network mechanism used in processing client requests to be shared by the multiple virtual host systems.

Brief Summary Text (23):
The virtual network mechanism contains a mapping component which maps the different IP address portions in a predetermined manner. The mechanism then reintroduces the packet containing the mapped IP address onto the interface of the IP module just as if it had been received from the other network. In greater detail, the IP destination address is mapped to now identify the host system in lieu of a specific hosted system and to replace the "well-known" port number with non-well-known port identifier of the services application program/server (e.g. FTP application server). Additionally, the mapping unit substitutes a virtual host address for the IP source address of the requesting client application program on the remote host system so that any reply packets provided by the application services server in response to the request are automatically directed back to the virtual network mechanism.

Drawing Description Text (3):
FIG. 2 is a simplified system block diagram illustrating the use of the virtual network of the present invention in an internetwork.

Drawing Description Text (4):
FIG. 3 is a diagram illustrating the positioning of the virtual network mechanism within a layered communication network, according to the teachings of the present invention.

Drawing Description Text (5):
FIG. 4 is a block diagram of the virtual network mechanism of the present invention.

Drawing Description Text (6):
FIG. 5 illustrates in greater detail, the different structures utilized by the virtual network mechanism of the present invention.

Detailed Description Text (2):
FIGS. 1a and 1b collectively constitute a block diagram of a host system 54 which incorporates the virtual network mechanism of the present invention. As shown, the system 54 includes a hardware platform 56 which contains the hardware elements such

as a central processing unit 58a, a main memory 58b and a number of input/output peripheral devices 58c and a communications facility such as an Ethernet local area network (LAN) 58d for connecting system 54 to other processing systems via standard communication network facilities.

Detailed Description Text (31):
Virtual Network Mechanism

Detailed Description Text (32):
According to the teachings of the present invention, the operating system level 64 also includes a virtual network (VNET) mechanism 100 which operatively couples to the TCP/IP network protocol stack facility 99 in the same manner as the network interface associated with the network driver and LAN 58d couples to facility 99 as explained in detail herein. The VNET mechanism 100 also couples to a plurality of sets of structures represented by block 102 located in host system memory which are used to process client requests received via facility 99 directed to a plurality of virtual host systems/hosted systems.

Detailed Description Text (34):
FIG. 2 is a simplified block diagram of a portion of a internetwork system 10 which discloses in greater detail, how the VNET mechanism 100 of the present invention is incorporated into the host system of FIG. 1. As seen from the Figure, only the components relevant to describing the teachings of the present invention are depicted in FIG. 2. As indicated, the VNET mechanism 100 functionally represents a plurality of virtual host systems ve0 through ve3 running a corresponding number of emulating hosted operating systems, such as emulator 80. In the preferred embodiment, each virtual host system connects to a local area network which corresponds to the virtual LAN of block 100. As described herein, the network structure of the emulated system in terms of IP address is incorporated into the host system 54 by configuring the virtual network mechanism 100 into the host system as described herein.

Detailed Description Text (35):
As shown, the mechanism 100 includes a virtual network interface portion 100-2. In many respects, this interface is functionally similar to the network interface labeled 58d connected to the physical local area network (LAN) 18. In addition to the LAN, the interface 58d includes the standard software routines (e.g. drivers) which provide a uniform interface to the Internet Protocol (IP) network layer. Thus, the interface performs all of the necessary communications between the IP layer and the physical LAN normally through an appropriate physical device handler. For the purposes of the present invention, the software portion of the network interface 58d may take the form of the AIX Network Interface Driver(s) described in standard IBM publications.

Detailed Description Text (36):
As described later herein, the virtual network interface 100-2 is also constructed to incorporate the same functionality as included in the network interface software of block 58d. In the case of an Ethernet LAN consisting of host machines which use the TCP/IP protocols, such as shown in FIG. 2, there are two types of addresses. One is the 32 bit Internet address and the other is the 48 bit Ethernet address. Typically, Ethernet addresses are assigned by the manufacturer of the interface board and are all unique. To determine the Ethernet address which corresponds the host system having a particular IP address, an Internet Address Resolution Protocol (ARP) is used wherein a host is allowed to broadcast a special packet on the Ethernet that asks the host with a specified IP address to respond with its Ethernet address. The broadcasting host system then can store the response and maintain the mapping between the IP address and the Ethernet address for all future packets designating that IP address.

Detailed Description Text (38):

The present invention makes use of the routing capabilities of the IP module. A gateway determines the route of a packet by consulting a network routing table. In TCP/IP, routing can be one of two types. The first type is static routing which uses manual input to update the routing table. The second type is dynamic routing which uses routing daemons to update the routing table automatically when new information is received. Therefore, when the host system 20 desires to communicate with the virtual network mechanism 100, it utilizes a route command which allows a user on host system 20 to make manual entries into the network routing tables. In the preferred embodiment, a host system route command is used to statically configure a gateway for the virtual host system 100-4 connected to the virtual LAN of virtual network mechanism 100 to which the user on host system 20 wants to connect. The route command has the following format: route add -net network.sub.--address gateway.sub.-- address. When the operating system is rebooted, the gateways must be configured again. For a static or permanent configuration, gateways can be configured via the operating system configuration management system.

Detailed Description Text (39):
As shown in FIG. 2, the LAN 18 in addition to connecting to host system 54 also connects to another host system 20. When the virtual network mechanism 100 is configured into the system, it is viewed by the host system 54 as another network since it is constructed to have its own separate network interface. Each IP address includes a network ID field and a host lED. As indicated above, host systems which attach to two or more networks are "gateways." That is, a gateway has two or more network interfaces, one for each network with which its communicate regardless of network type.

Detailed Description Text (41):
Therefore, when host system 20 adds the virtual network IP address to its network routing table, the same routing information is also passed to host system 54 through static or dynamic routing and entered into the network routing tables utilized by the IP module of the host system 54 on which the virtual network mechanism 100 resides. Accordingly, as described later herein, the IP module automatically routes those IP packets/designating the virtual LAN to virtual network mechanism 100.

Detailed Description Text (42):
FIG. 3-Virtual Network Mechanism Location

Detailed Description Text (43):
FIG. 3 illustrates in diagrammatic form, the positioning of the virtual network mechanism 100 according to the present invention, relative to the TCP/IP conceptual layered organization. As indicated in FIG. 3, the VNET mechanism 100 directly couples to the IP layer so that it looks like another network interface to the host operating system TCP/IP protocol stack. The application layer is the level at which the TCP/IP application programs or user processes operate/reside. The several application programs provided by almost every TCP/IP implementation include FTP and Telnet which were discussed above.

Detailed Description Text (49):
FIG. 4-Well-Known Port Virtual Network Mechanism Block Diagram

Detailed Description Text (50):
FIG. 4 illustrates the various parts of the Virtual Network Mechanism 100. As shown, the mechanism 100 includes the components 100-2 through 100-14 which operatively connect as shown. The IP interface component block 100-2 represents the various interface routines utilized by the different sets of structures corresponding to the virtual host systems ve0 through ve3. In the preferred embodiment, the interface table structure 100-2 defines one of the three types of physical interfaces. For the purpose of the present invention, the interface 100-2 conforms to the type of network interface utilized within the AIX operating system.

Generally, this type of interface accepts output packet of a specified maximum length, and provides input packets received from its medium to higher level routines.

Detailed Description Text (53):
The ifnet structure has the format indicated in FIG. 7c. The functions of the ifnet structure include loading and initializing, communicating with the IP network layer, communicating with device handler software, translating an IP address to a hardware address for the underlying device driver software, handling ifnet specific ioctl calls and terminating and unloading. The present invention makes use of this same type of network structure mechanism utilized by the host operating system for a physical network interface unit which eliminates the need to introduce any additional network structures or software to be associated with the virtual network mechanism 100.

Detailed Description Text (54):
As indicated in FIG. 7c, the ifnet structure contains a number of different fields, only some of which are utilized by the virtual network mechanism 100. A first field is a name field (if.sub.-- name) which identifies the virtual host (i.e., ve0, vel, ve2 or ve3). A second field is a unit field (if.sub.-- unit) which is an integer used to locate the virtual host system control structure associated with the virtual host system (i.e. ve.sub.-- softc). The ifnet interface structure also includes interface property fields such as the flags field (if.sub.-- flags) which is used to indicate the state of the interface/virtual host system (e.g. an IZFF.sub.-- UP state indicating that the interface/virtual host is up, an IFF.sub.-- RUNNING state indicating that the interface/virtual host is running which allocates resources), an ifaddr structure which contains information about one interface address which is a pointer to a linked list of addresses used by the IP module to locate all of the network interfaces of a given address family on the host system (e.g. Ethernet interface 58d), interface routines fields which identify the different routines used by an attached interface (e.g. if.sub.-- init, if.sub.-- output, if.sub.-- ioctl) and interface statistics fields.

Detailed Description Text (56):
As seen from FIG. 5, each ve.sub.-- softc structure includes a number of different fields and structure designated struct arpcom through virtual IP address. The structure arpcom defines a network common structure which is shared by the mechanism 100 and the so-called address resolution code which can be viewed as standard. The if.sub.-- name field is used to define the virtual host system name (e.g. ve0) while the ve.sub.-- flags field is used for storing a private flag. The state field defines the state of the virtual host system while the client.sub.-- count field defines the number of different client processes in the table. The client table pointer field defines the address of the first client table as indicated in FIG. 5. The local IP address field is used for storing a commonly used local host IP address value while the virtual host IP address field is used for storing a unique virtual host IP address value. By using a common local host IP address, this eliminates the need to replicate the software routines of the virtual network interface 100-2 of FIG. 2.

Detailed Description Text (61):
The initialization component 100-4 includes a number of routines for performing the operations required for initializing the virtual network mechanism 100 and the sets of virtual host control structures inet, ve.sub.-- softc and client table control structures associated with each of the virtual host systems ve0-ve3.

Detailed Description Text (63):
With reference to FIGS. 1 through 8, the operation of the preferred embodiment of the virtual network mechanism 100 of the present invention will now be described. By way of example, it is assumed that a number of client user processes running on the remote host system 20 of FIG. 2 want to utilize the emulated system FTP

services application program 22 running on host system 54. In accordance with the teachings of the present invention, host system 54 is configured to attach to the IP layer, a plurality of network interfaces, one for each emulating hosted operating system/virtual host which are utilized by virtual network mechanism 100 to communicate with the IP layer. When so configured, the virtual network mechanism 100 operates with the different sets of structures, each of which has the local host IP address and its own virtual host IP address.

Detailed Description Text (65):
By way of example, it is assumed that the virtual host 100-4a has an IP address value of "215.65.43.2" wherein the value "215.65.43" again designates the network address of the virtual LAN and the value "2" designates the virtual host address of the emulated system/virtual host 100-4 which connects to the virtual LAN. Each of the other virtual host systems ve1 through ve3 has IP address values which corresponds to the incremented IP address of its local host system (e.g. ve1, ve2, ve3) as for example, address values "215.65.43.3" through "215.65.43.5." Again, the value "215.65.43" designates the network address of the virtual LAN and the values "3" through "5" designate the virtual host IP addresses of the virtual host systems 100-4b through 100-4d connected to the virtual LAN. It will be understood that the IP virtual addresses and the network LAN could have other values.

Detailed Description Text (69):
Additionally, the host system 54 must also configure the local host IP address for virtual network mechanism 100-2 to communicate with the virtual host systems ve0-ve3. According to the present invention, this may be done by means of separate "VIRNET" directives included in the hosted system (emulator) configuration file clm.sub.-- x file. Each VIRNET directive has the following format: VIRNET ve(n) [ctl.sub.-- args] wherein the first argument "ve(n)" specifies the particular virtual network interface mechanism 100 according to "n" which has the values "0" through "3."

Detailed Description Text (70):
The remaining arguments include an address, up and down arguments. The "address" argument corresponds to either a host name or an IP address in the standard dotted decimal notation. The address used for this argument is assigned to the host side of the virtual network interface mechanism 100 (i.e., local host interface 100-2). This address is automatically incremented by one to create the IP address for the first virtual host system ve0 connected to the virtual LAN on the opposite side of virtual network mechanism 100-2. The "up" argument is used to activate the virtual network interface mechanism 100-2 while the "down" argument is used to deactivate the virtual network interface mechanism 100-2.

Detailed Description Text (71):
When a VIRNET directive is used in this example to configure the first virtual host 100-4a which connects the virtual network mechanism, the directive has the following form: VIRNET ve0 215.65.43.1 up wherein the value "215.65.43.1" corresponds to the local host IP address and "215.65.43.2" corresponds to the virtual host IP and "up" specifies the activation of the mechanism 100. The VIRNET directive is entered into the hosted operating system (emulator) clm.sub.-- x file and is used for loading and configuring the virtual network mechanism 100 software into the operating system kernel of host system 54. Other VIRNET directives having similar forms can be used to configure other ones of the virtual hosts ve1 through ve3. In the convention used by the present invention, it is not necessary to again setup the local host IP address since it was previously configured when the first virtual host ve0 was configured. As indicated, each virtual host system uses the same local host IP address which allows the use of the same software routines included as part of virtual network interface 100-2.

Detailed Description Text (72):
If a virtual host system is not configured via directive, it can be started from an

operating system command line using a special command which serves the same function as the VIRNET directive. This command has the format: hvx.sub.-- vecfg re (n) [ctl.sub.-- args] wherein "n" is used to designate the specific virtual host system (e.g. ve0, ve1, etc.). The arguments ctl.sub.-- args are the same as those of the VINET directive. The command can be used at any time to activate the virtual network mechanism 100-4a or change its parameters. In the present example, the command used to configure mechanism 100 has the following form: hvx.sub.-- vecfg ve0 215.65.43.1 up. The command configures and starts virtual network mechanism 100 with an IP address of 215.65.43.1. As previously mentioned, this address is automatically incremented to establish the virtual host IP address of 215.43.2 for the first virtual host system ve0 running the emulating hosted operating system. The other virtual host systems are similarly configured but without performing any further increment operation.

Detailed Description Text (76):
Next, as indicated in block 704, the host system 54 builds the ifnet structure of FIG. 7c for the virtual host system 100-4a. It initializes its fields so that it contains with the addresses of the interface functions/routines (i.e., if.sub.-- output, if.sub.-- ioctl and if.sub.-- reset) utilized by the virtual network mechanism 100. Additionally, the appropriate value designating the type of interface which is "Ethernet" in the present example, is also loaded into the structure. As indicated in block 705, system 54 saves the unit number value identifying the virtual host system ve0 in the if.sub.-- unit portion of the associated ifnet structure.

Detailed Description Text (77):
Next, as indicated in block 706, the host system calls the if.sub.-- attach kernel services of the AIX network interface device software layer which adds the virtual network mechanism 100 as another network interface to the system wide network interface list. That is, the configured ifnet and ve.sub.-- softc structures are properly registered. Also, as indicated in block 708, the host system turns on the timer function which provides an arbitrary value (e.g. 20 minute) time interval to clean out stale client table entries. This completes this portion of the initialization sequence of block 602.

Detailed Description Text (79):
In the preferred embodiment, as discussed above, the virtual host IP address for virtual host ve0 is generated by adding one to the local host IP address (i.e., 215.65.43.1). The resulting value (i.e., 215.65.43.2) for virtual host 100-4a is saved in the virtual host IP address portion of the control structure ve.sub.-- softc of FIG. 5. Next, as indicated in block 724, the host system sets the IFF.sub.-- UP flag of the if.sub.-- flags field of the ifnet structure for the virtual network mechanism 100 to a state which indicates that the interface is "up."

Detailed Description Text (80):
As seen from FIG. 7b, a second type of ioctl command (i.e.,SIOCSIFFLAGS) is executed which sets the interface IFF.sub.-- RUNNING flag to indicate that the interface is "running." This enables the allocation of resources by the system which places the virtual network mechanism 100 in an operative (running) state as indicated in block 730. The above sequence of operations of FIGS. 7a and To is repeated for each of the other configured virtual host systems 100-4b through 100-4d.

Detailed Description Text (81):
Referring to FIG. 6, once initialization has been completed, the virtual network mechanism 100 is ready to receive packets from remote system 20 specifying any one of the virtual host systems ve0-ve3. As discussed above, the remote system 20 sends packets to the host having IP address 215.65.43.1 via the IP module of local host system 54 which operates as a "gateway." That is, the IP module receives each data

packet and determines that the data packet should be routed to one of the virtual host systems ve0-ve3 through the virtual network interface as specified by the local host IP address.

Detailed Description Text (84):
As indicated above, it may be desirable to utilize multiple virtual host systems to take advantage of multiple processor resources of a multiprocessor system. In such cases, it is only necessary to provide one type of protocol, such as a specific Ethernet protocol. In other instances, multiple virtual host systems may be used to operate in conjunction with different types of physical networks, such as Ethernet, Token Ring, FDDI, etc. or operate in conjunction with different protocols of a specific type of physical network, such as Ethernet. From an implementation point of view, it may be desirable to utilize a separate virtual LAN for each different physical network media (e.g. Ethernet, Token Ring, FDDI). In this instance, it is necessary to replicate virtual network interface 100-2 within each virtual LAN and assign each such network interface, a different local host IP address value.

Detailed Description Text (104):
From FIG. 8 and the above descriptions, it is seen how the mechanism of the present invention allows host and hosted system application programs executable by multiple emulating hosted operating systems/virtual host systems sharing a single host TCP/IP communications network stack to use the same well-known port without having to make any changes in client application programs. The mechanism of the present invention by operating below the IP layer of a network stack is able to take advantage of the routing capabilities of the IP layer/module. This minimizes the amount of software required to be added to the host operating system facilities in incorporating the virtual network mechanism of the present invention.

Current US Original Classification (1):
709/250

CLAIMS:

1. A method which allows a local host system to share a network software facility of the local host system operating system between a number of application servers operating under the host operating system and a corresponding number of application servers operating under components of a plurality of hosted operating systems running under control of the local host operating system, the local host system being coupled to at least one remote host system through a local area network (LAN) and an internetwork, the network software facility being coupled to a network interface unit which includes interfacing hardware and software for connecting the local host system to the LAN for communicating with the remote host system using a standard communications network protocol which is characterized by assigning different station address identifier values to each host system and well-known services function identifier values to the different data communications application servers associated with local host system and hosted operating systems so that servers performing the same service function are assigned the same well-known services function identifier value for directing incoming packets sent by the remote host system to the appropriate application server, said method comprising the steps of:

(a) configuring a virtual network mechanism within the local host operating system to be operatively coupled to the host operating system network software facility through a plurality of network interface structures to function as a virtual LAN connected to a plurality of virtual host systems running the hosted operating system with each virtual host system operating as if it contained its own network software facility;

(b) preallocating memory and initializing a different set of structures in preallocated memory for each of the plurality of virtual host systems which operate

in conjunction with the virtual network mechanism and the plurality of hosted operating systems, each different set of structures containing a unique unit number identifying the virtual host systems associated therewith and a unique IP address designating the particular virtual host system within the virtual LAN;

(c) mapping predetermined portions of each incoming packet by the virtual network mechanism sent by the remote host system and received from the local host communications network software facility by changing the station address identifier value of each incoming packet to specify the local host system as a destination and the particular virtual host system as a source of the packet for returning any reply packet and changing the well-known services identifier value to a virtual host identifier value so that the packet received from the virtual network mechanism is directed by the network software facility to the appropriate application server of the designated one of the plurality of hosted operating system for processing; and,

(d) remapping the predetermined portions of each outgoing reply packet sent by a hosted system application server through the network software facility to the particular virtual host system by restoring the remote host station address identifier and well-known service identifier values so each outgoing reply packet sent by the virtual network mechanism to the internetwork appears to the remote host system as a reply packet to the communication between the remote host system and the hosted system application server as if the server had been reached through the LAN using the originally sent station address assigned to the particular hosted operating system with the well-known services identifier value.

2. The method of claim 1 wherein the virtual network mechanism includes interfacing software similar to the network interface unit and a common set of software routines utilized by each of the plurality of virtual host systems.

3. The method of claim 2 wherein the network software facility includes a TCP/IP protocol stack containing TCP and IP layers and the virtual network mechanism utilizes the network routing capabilities of the IP layer.

13. The method of claim 8 wherein the second structure contains a predetermined number of fields, a first field designating the name of the virtual host system, a second field for storing the state of the virtual host system, a third field for maintaining a count of the number of different client entries being managed by the virtual network mechanism, fourth and fifth fields for storing the common local host and unique virtual host station address identifier values respectively and a sixth field for storing a client pointer value for accessing the first client table structure generated by the virtual host system.

18. A virtual network mechanism which allows a local host system to share a network software facility of the local host system operating system between a number of data communications application servers operating under the host operating system and a corresponding number of application servers operating under components of a plurality of hosted operating systems running under control of the local host operating system, the local host system being coupled to at least one remote host system through a local area network (LAIN) and an internetwork, the network software facility being coupled to a network interface unit which includes interfacing hardware and software for connecting the local host system to the LAN for communicating with the remote host system using a standard communications network protocol which is characterized by assigning different station address identifier values to each host system such that the local host system and hosted operating systems are assigned different station addresses and well-known services function identifier values to the different data communications application servers associated with local host system and each of the plurality of hosted operating systems so that servers performing the same service function are assigned the same well-known services function identifier value for directing incoming communication

data packets sent by the remote host system to the appropriate communications application server running on the particular hosted operating system, said mechanism comprising:

(a) an interface component configured within the local host operating system to operatively couple the virtual network mechanism to the host operating system communications network software facility as a virtual LAN connected to a plurality of virtual host systems which are the components of the plurality of hosted operating systems;

(b) an initialization component for preallocating and initializing a different set of structures for each of the plurality of virtual host systems which operate in conjunction with the virtual network mechanism and the plurality of hosted operating systems, each different set of structures being initialized to contain a unique number value identifying a particular one of the virtual host systems and a unique IP address designating the virtual host system on the virtual LAN;

(c) a first mapping component coupled to the interface component for mapping predetermined portions of each incoming packet sent by the remote host system and received from the interface component through the local host network software facility so that the station address identifier value of each incoming packet is changed to specify the common local host system as a destination and the particular virtual host system as a source of the packet for processing each reply packet and the well-known services identifier value is changed to a virtual identifier value so that the packet received from the virtual network mechanism is directed by the network software facility to the appropriate application server of the designated hosted operating system for processing; and,

(d) a second mapping component for mapping the predetermined portions of each outgoing reply packet sent by a hosted system communications application server through the network software facility to the interface component by restoring the remote host station address identifier and well-known service identifier values so each outgoing reply packet sent by the virtual network mechanism to the internetwork appears to the remote host system as a reply packet to the communication initiated by a client application program running on the remote host system and the hosted system application server as if the server had been accessed through the LAN using the originally sent station address assigned to the particular hosted operating system by the well-known services identifier value.

20. The mechanism of claim 19 wherein the first structure is an interface network structure utilized by the host operating system to communicate with the virtual host system network facility and the second structure is a software control structure which the virtual host system uses to manage packet processing for each of the client application programs running on the remote host system., the software control structure containing a predetermined number of fields, a first field designating the name of the virtual host system, a second field for storing the state of the virtual host system, a third field for maintaining a count of the number of different client entries being managed by the virtual network mechanism, fourth and fifth fields for storing the common local host and unique virtual host station address identifier values respectively and a sixth field for storing a client pointer value for accessing the first client table structure generated by the virtual host system.

Previous Doc Next Doc Go to Doc#

US005734865A

United States Patent [19]

Yu

[11] Patent Number: 5,734,865

[45] Date of Patent: Mar. 31, 1998

[54] **VIRTUAL LOCAL AREA NETWORK WELL-KNOWN PORT ROUTING MECHANISM FOR MULT--EMULATORS IN AN OPEN SYSTEM ENVIRONMENT**

[75] Inventor: **Kin C. Yu**, Burlington, Mass.

[73] Assignee: **Bull HN Information Systems Inc.**, Billerica, Mass.

[21] Appl. No.: **495,160**

[22] Filed: **Jun. 27, 1995**

Related U.S. Application Data

[63] Continuation-in-part of Ser. No. 473,476, Jun. 7, 1995, Pat. No. 5,636,371.

[51] Int. Cl.^{6} **G06F 13/00**; G06F 15/163; G06F 15/177

[52] U.S. Cl. **395/500**; 395/200.02; 395/681; 395/682; 395/684; 364/242.94; 364/242.95; 364/242.96; 364/DIG. 1; 370/254

[58] Field of Search .... 395/500, 200.02, 395/680, 684, 575, 681, 682; 370/230, 231, 399, 404, 397, 254; 364/242.94, 242.95, 242.96, DIG. 1

[56] **References Cited**

U.S. PATENT DOCUMENTS

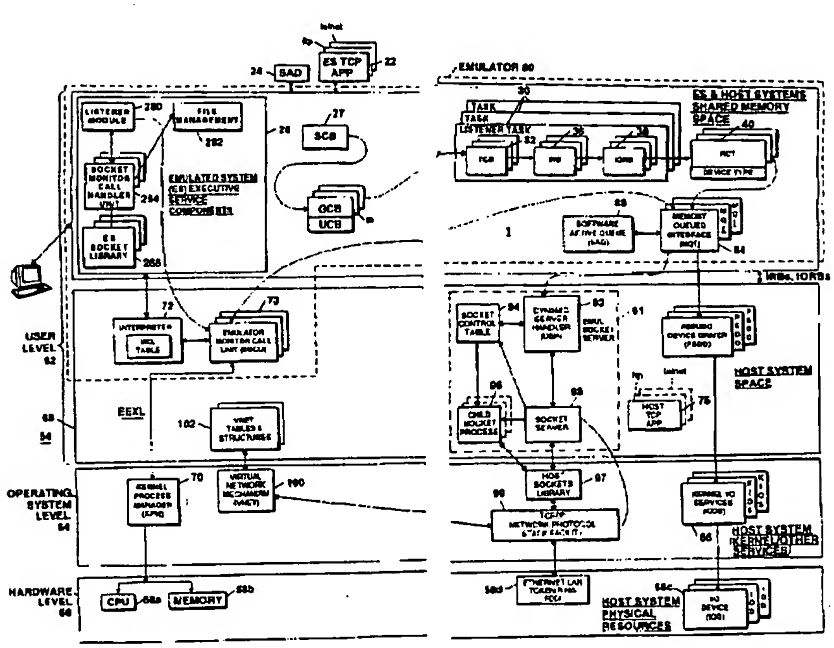
5,111,384	5/1992	Aslanian	395/575
5,271,010	12/1993	Miyake et al.	370/392
5,313,454	5/1994	Bustini et al.	370/231
5,339,435	8/1994	Lubkin et al.	395/701

Primary Examiner—Kevin J. Teska
Assistant Examiner—Thai Phan
Attorney, Agent, or Firm—Faith F. Driscoll; John S. Solakian

[57] **ABSTRACT**

A local host data processing system operating under the control of a local host operating system includes components of multiple emulating hosted operating systems. The host operating system further include a TCP/IP network protocol stack which couples to the communications facilities of the host system connected to a local area network for communicating with a number of remote host systems. Host and hosted operating systems share the same TCP/IP network protocol stack. A virtual network mechanism is configured within the local host system to be operatively coupled to the host network protocol stack and provide access to well-known port application programs. When so configured, the mechanism functions as another LAN to which multiple virtual host systems are attached for executing applications under control of the emulating hosted operating systems. The mechanism transforms the well-known port identifier of each inbound packet into a non-well-known port identifier in addition to other station address identifier fields. It then redirects the transformed packet back to the IP layer of the stack for transfer to the appropriate well-known port application program being run by the hosted operating system of the particular virtual host system. The mechanism reverses this operation for each reply packet which it redirects back to the IP layer for forwarding to the remote system. This eliminates the need to specify additional protocol stacks and to provide additional communication hardware facilities for handling multiple instances of well-known port applications programs running on the different virtual host/multiple hosted operating systems.

20 Claims, 15 Drawing Sheets



Hit List

Clear | Generate Collection | Print | Fwd Refs | Bkwd Refs | Generate OACS

Search Results - Record(s) 1 through 1 of 1 returned.

☐ 1. Document ID: US 6693878 B1

L3: Entry 1 of 1 File: USPT Feb 17, 2004

DOCUMENT-IDENTIFIER: US 6693878 B1
TITLE: Technique and apparatus for using node ID as virtual private network (VPN) identifiers

Detailed Description Text (20):
The SID (service ID) is a DOCSIS MAC (layer 2) device and service identification which is assigned by the CMTS to each cable modem in the network. Because the SID may be expressed using a fewer number of bits than the corresponding MAC address for a particular cable modem, the SID is the preferred identifier for communications between a cable modem and the CMTS. In a specific embodiment of the present invention, the service ID or SID associated with a particular cable modem is also used as a VPN or sub-interface identifier whereby each SID corresponding to a particular cable modem is linked to information identifying the particular VPN/sub-interface to which that cable modem is to be mapped. In this way, VPN traffic flows may be separated at the CMTS based on the SID tag associated with a particular packet.

Detailed Description Text (24):
FIG. 6 shows an example of a MAC address/IP address mapping table 600 in accordance with a specific embodiment of the present invention. Table 600 of FIG. 6 has been provisioned to include a list of MAC addresses 610 corresponding to at least a portion of cable modems in the cable network. Each MAC address in table 600 is also provisioned to be associated with a specific range of IP addresses 614. When the DHCP server receives an IP address request from a specific cable modem, the server uses the MAC address of that cable modem to determine the appropriate address range from which to select and assign an IP address to the requesting CM, based upon Table 600. In a specific embodiment, the specific IP address range assigned to each MAC address in table 600 inherently corresponds to a particular VPN. Thus, for example, the cable modems corresponding to MAC addresses 1, 2, and 3, (601, 603, 605) of Table 600 have each been provisioned to be associated with the virtual private network VPN1, and therefore will each be assigned an IP address selected from the IP address range designated by "range 1". Similarly, the cable modems associated with MAC addresses 21, 22, 23 have each been provisioned to be associated with the virtual private network VPN2. Therefore, these cable modems and all other cable modems which are members of VPN2 will be assigned an IP address selected from the IP address range designated by "range 2" of Table 600.

Detailed Description Text (26):
Furthermore, as stated previously, it is possible for a cable modem to have more than one MAC address. In a specific embodiment of the present invention, each MAC address may be associated with only one specific VPN. Therefore, if the cable modem is to be associated with two VPN(s) for example, then it is preferable that the cable modem have two separate MAC addresses, where each MAC address is associated

with one of the VPNs of which the cable modem is a member. In this way, a first MAC address may be assigned an IP address corresponding to the IP address range associated with the first VPN, and the second MAC address may be assigned an IP address from the range corresponding to the second VPN. Moreover, in situations where the cable modem is associated with more than one VPN, the modem will typically be serving more than one device which sits behind the modem, where each device may be a member of separate VPN. This is shown, for example, in FIG. 3.

Detailed Description Text (28):
When a cable modem serves different devices which are members of different VPNs, the cable modem will be associated with all of the VPNs of all the devices the modem serves. In the example of FIG. 3, the cable modem CM2 is associated with both VPN1 and VPN2. In order to provide this multiple-VPN association, the cable modem must be able to furnish two different MAC addresses to the Head End, one MAC address for each separate VPN. In a specific embodiment, the MAC addresses are provided by each of the devices behind the cable modem. Thus, for example, the PC1 device 314 will supply a first MAC address to cable modem 304 when requesting an IP address, and PC device PC2 (316) will provide a second MAC address to the cable modem 304 when requesting its IP address. Cable modem 304 passes each of the IP address requests onto the CMTS, as shown, for example, in FIG. 4.

Detailed Description Text (58):
Generally, the techniques of the present invention may be implemented on software and/or hardware. For example, they can be implemented in an operating system kernel, in a separate user process, in a library package bound into network applications, on a specially constructed machine, or on a network interface card. In a specific embodiment of this invention, the methods of the present invention are implemented in software such as an operating system or in an application running on an operating system.

Current US Cross Reference Classification (6):
709/238

Full | Title | Citation | Front | Review | Classification | Date | Reference | | | Claims | KWIC | Draw De

Clear | Generate Collection | Print | Fwd Refs | Bkwd Refs | Generate OACS

Term	Documents
ISOLAT$	0
ISOLAT	28
ISOLATA	3
ISOLATABILITY	30
ISOLATABLE	2009
ISOLATABLE-TYPE	1
ISOLATABLE-VOLUME	1
ISOLATABLITY	1
ISOLATABLY	9
ISOLATAD	3
ISOLATAED	2

(L2 AND (ISOLAT$ OR SEPARAT$)).USPT.	1

There are more results than shown above. Click here to view the entire set.

Display Format: KWIC Change Format

Previous Page Next Page Go to Doc#

US006693878B1

(12) **United States Patent**
Daruwalla et al.

(10) **Patent No.: US 6,693,878 B1**
(45) **Date of Patent: Feb. 17, 2004**

(54) **TECHNIQUE AND APPARATUS FOR USING NODE ID AS VIRTUAL PRIVATE NETWORK (VPN) IDENTIFIERS**

(75) Inventors: **Feisal Y. Daruwalla**, Fremont, CA (US); **James R. Forster**, Los Altos, CA (US); **Mark W. Litwack**, West Chester, PA (US)

(73) Assignee: **Cisco Technology, Inc.**, San Jose, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **09/418,469**

(22) Filed: **Oct. 15, 1999**

(51) **Int. Cl.**[7] **H04L 12/56**

(52) **U.S. Cl.** **370/235**; 370/392; 370/395.31; 370/395.52; 370/397; 370/409; 709/238

(58) **Field of Search** 370/231, 235, 370/236, 352, 389, 392, 395.1, 396, 397, 395.3, 395.31, 395.5, 395.52, 400, 401, 409; 709/201, 203, 227, 232, 238, 243, 249

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,768,271 A * 6/1998 Seid et al. 370/389
6,069,889 A * 5/2000 Feldman et al. 370/351
6,137,793 A * 10/2000 Gorman et al. 370/360
6,205,488 B1 * 3/2001 Casey et al. 709/238
6,339,595 B1 * 1/2002 Rekhter et al. 370/392
6,438,127 B1 * 8/2002 Le Goff et al. 370/389
6,463,061 B1 * 10/2002 Rekhter et al. 370/392

OTHER PUBLICATIONS

Rosen, et al., Request for Comments: 2547: BGP/MPLS VPNs, Internet Engineering Task Force, Mar. 1999.
Fox, et al., "Request for Comments: 2685: Virtual Private Networks Identifier," Internet Engineering Task Force, Sep. 1999.
Anonymous, "Deploying Cable-Based Access and Intranet Virtual Private Networks," Cisco Systems, Inc., Jun. 1999.

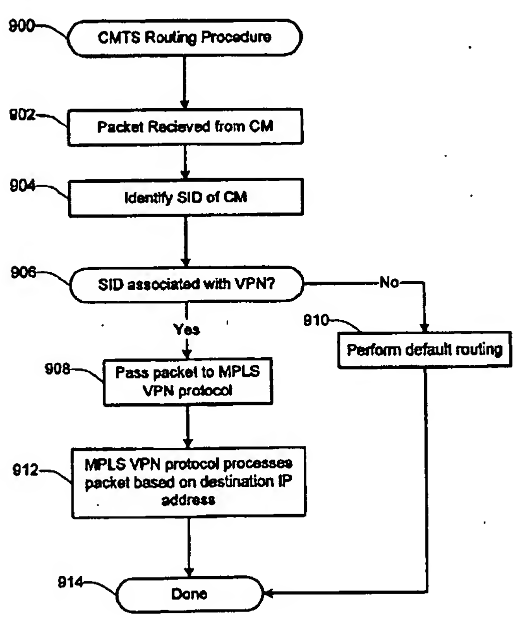
* cited by examiner

Primary Examiner—Alpus H. Hsu
(74) *Attorney, Agent, or Firm*—Beyer, Weaver & Thomas, LLP.

(57) **ABSTRACT**

A technique is provided for managing VPN packet flows over shared access data networks. Each node in the shared access network typically has an identifier or ID associated with it which is used at a Head End of the shared access network to uniquely identify that particular node from the other nodes in the network. According to the technique of the present invention, the node ID may be used at the Head End of the network to identify not only the corresponding node, but also to identify any virtual private networks (VPNs) of which the corresponding node is a member. Using the technique of the present invention, nodes which are members of the same VPN within a shared access network may exchange packets in a manner which does not require the packets to be routed outside the shared access network.

79 Claims, 12 Drawing Sheets



L2: Entry 2 of 8 File: USPT Mar 4, 2003

DOCUMENT-IDENTIFIER: US 6529517 B2
**** See image for Certificate of Correction ****
TITLE: Router for which a logical network address which is not unique to the router is the gateway address in default routing table entries

Detailed Description Text (91):
Moreover, the techniques described herein for dynamically assigning IP addresses to hosts will work with any kind of logical network addresses, including, for example, virtual circuit numbers. Similarly, the techniques described for dynamically assigning <channel,pipe,linkID> triples to RF modems can be used equally well to dynamically assign any kind of link-level address. The techniques will also work with any technique for subdividing the bandwidth of the unidirectional connection among a number of modems.

Current US Cross Reference Classification (1):
709/238

US006529517B2

(12) United States Patent
Hrastar et al.

(10) Patent No.: US 6,529,517 B2
(45) Date of Patent: *Mar. 4, 2003

(54) ROUTER FOR WHICH A LOGICAL NETWORK ADDRESS WHICH IS NOT UNIQUE TO THE ROUTER IS THE GATEWAY ADDRESS IN DEFAULT ROUTING TABLE ENTRIES

(75) Inventors: Scott E. Hrastar, Duluth, GA (US); George Horkan Smith, Atlanta, GA (US)

(73) Assignee: Scientific-Atlanta, Inc., Lawrenceville, GA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

This patent is subject to a terminal disclaimer.

(21) Appl. No.: 09/767,926

(22) Filed: Jan. 23, 2001

(65) Prior Publication Data

US 2001/0012297 A1 Aug. 9, 2001

Related U.S. Application Data

(63) Continuation of application No. 08/840,304, filed on Apr. 11, 1997, now Pat. No. 6,249,523.

(60) Provisional application No. 60/035,618, filed on Jan. 17, 1997.

(51) Int. Cl.[7] H04L 12/28; H04L 12/56

(52) U.S. Cl. 370/401; 709/238

(58) Field of Search 370/389, 400, 370/401, 402, 392, 352, 355; 709/238, 239

(56) References Cited

U.S. PATENT DOCUMENTS

3,985,962 A 10/1976 Jones et al.

(List continued on next page.)

OTHER PUBLICATIONS

Data-Over Cable Service Interface Specifications; Operations Support System Interface Specification Baseline Privacy Interface MIB; SP-OSSI-BPI-I01-980331; 1997 & 1998; MCNS Holdings, LP; pp. 1-33.

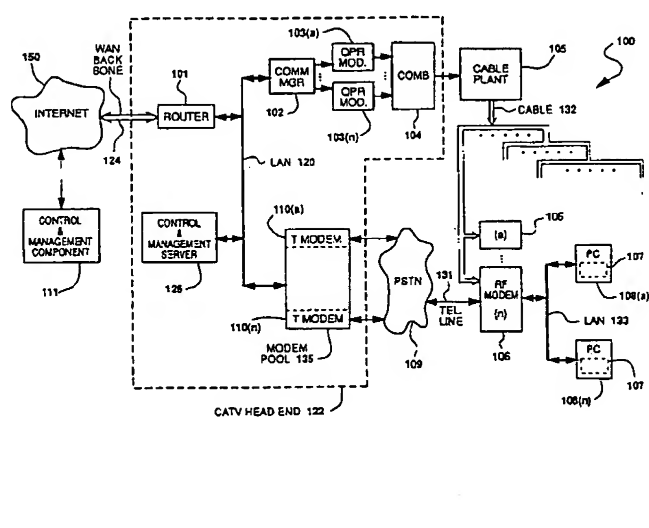
(List continued on next page.)

Primary Examiner—Ajit Patel
Assistant Examiner—Bob A. Phunkulh
(74) *Attorney, Agent, or Firm*—Kenneth M. Massaroni; Hubert J. Barnhardt, III; Shelley L. Couturier

(57) ABSTRACT

An asymmetrical network for coupling customer-premises Internet hosts such as personal computers to the Internet. The head end of a CATV system has a high-bandwidth connection to the Internet. The down link connecting the personal computers to the Internet is the cables provided by the CATV system; the up link is a telephone connection to the head end. A router is connected to the down link by means of a RF modem, to the up link by means of an analog modem, and to a LAN which is connected to the Pcs. The router routes IP packets for the hosts that are received on the CATV cable to the hosts via the LAN; it routes IP packets from the hosts that are destined for the Internet to the head end via the telephone line. The asymmetrical network conserves IP addresses and addresses on the CATV cable by dynamically allocating the IP addresses for an RF modem's hosts and an address on the CATV cable for the RF modem in response to a request made by the RF modem via the telephone line. It further saves IP addresses by assigning a non-unique IP address to the router for use inside the LAN. When the CATV system fails, the asymmetrical network automatically begins to use the telephone line as both the up link and the down link, and when the CATV system is restored, the asymmetrical network automatically returns to using the CATV cable as the down link and the telephone line as the up link. A further feature of the asymmetrical network is that the head end components and the RF modem have IP addresses, so that standard TCP/IP protocols can be used to control the asymmetrical network.

19 Claims, 10 Drawing Sheets



L9	L7 and (assign$ with IP with address$ with virtual with network$)	10	L9
L8	L7 and (assign$ with IP with address$)	157	L8
L7	L1 and (virtual adj2 network$)	826	L7
L6	L4 and (virtual with network$)	0	L6
L5	L4 and (assign$ with IP with address$)	1	L5
L4	5978568.pn.	1	L4
L3	L2 and (isolat$ or separat$)	1	L3
L2	L1 and (assign$ IP with address$ with virtual with network$)	8	L2
L1	709/$.ccls.	17870	L1

END OF SEARCH HISTORY

ABRA

Hit List

Clear | Generate Collection | Print | Fwd Refs | Bkwd Refs
Generate OACS

Search Results - Record(s) 1 through 1 of 1 returned.

1. Document ID: US 5978568 A

L5: Entry 1 of 1 File: USPT Nov 2, 1999

DOCUMENT-IDENTIFIER: US 5978568 A
TITLE: Method and apparatus for resolving network users to network computers

Abstract Text (1):
A naming service manager (46) is provided for resolving mapping information regarding a plurality of computers connected to a local area network (LAN 24) and their users. The naming service manager 46 collects mapping information, i.e., user login names, domain names, computer names and IP addresses, from a plurality of naming service agents 50 located on the LAN (24) and correlates the mapping information into a current computer-to-user assignment or "mapping" for each user of the LAN 24 and/or a current IP address-to-computer assignment or "mapping" for each computer connected to the LAN 24. The naming service manager (46) serves the correlated mapping information to a plurality of naming service applications (48) which process the correlated mapping information in accordance with their own needs and requirements.

Brief Summary Text (5):
Virtually any electronic device or computer equipped with the necessary hardware can be connected to a LAN or a WAN and hence, to the Internet. Each computer and device that is connected to the Internet has an Internet Protocol address ("IP address") that uniquely identifies the computer or device from all other computers and devices on the Internet. An IP address is comprised of four groups of numbers separated by decimals, for example, 165.113.245.2. Each computer connected to the Internet also has a computer name or "host name" that is assigned to the computer at its particular IP address. For example, the name "abc" can be assigned to a computer having the IP address 165.113.245.2. In turn, the computer name is embedded in what is known as a "fully qualified domain name" that uniquely identifies a computer connected to the Internet in a more user friendly manner. In its most generic form, a fully qualified domain name consists of three elements: The host name, the assigned domain name, and a top-level domain name. For example, a computer connected to the Internet may have the fully qualified domain name "abc.sequeltech.com", which includes the host name ("abc"), the domain name ("sequeltech"), and the and the top-level domain name ("com").

Brief Summary Text (6):
Fully qualified domain names are translated into numeric IP addresses and vice versa by domain name servers connected to the Internet. A domain name server is a computer containing software capable of responding to domain name inquiries and accessible on a full-time basis to other computers on the Internet. However, domain name servers do not maintain or keep track of individual computer name to IP address assignments within those networks making up the Internet. This function is provided by a domain controller server, i.e., a computer connected to a network that contains the software capable of keeping track of the computer name and IP

address of each computer connected to the network.

Brief Summary Text (7):
In the traditional network environment, network users are statically assigned or "mapped" to a particular computer at a particular IP address, computer name or domain name using various application program interfaces. Consequently, network resources, such as electronic mail, peripheral devices, CD-ROM libraries, etc. are made available, assigned or applied to users at the computers to which the users are assigned or mapped. However, if a user utilizes a computer connected to the network to which the user is not assigned or mapped, the network resources specifically intended for that user are not made available, assigned or applied to that computer. Rather, the network resources intended for that user remain with the computer to which the user is formally assigned, and the network resources applied to the computer currently being utilized by the user are those made available, assigned and applied to another user that is formally assigned to that computer.

Brief Summary Text (8):
Accordingly, a method and apparatus for resolving network users to network computers is needed so that as users log into and out of different computers connected to the network, the network resources intended for that user are applied to that user. The method and apparatus should be able to identify all computers currently being utilized by a specified user and identify which user is currently utilizing which computer. Further, the method and apparatus should keep track of which IP address is currently assigned to which computer. The method and apparatus should also allow for both dynamic and static resolution of network users to network computers. More specifically, the method and apparatus should provide for static or "permanent" user-to-computer and/or computer-to-IP address assignments as well as dynamic user-to-computer and/or computer-to-IP address assignments which change as users log into and out of computers connected to the network or as IP addresses change. Finally, the method and apparatus should be completely transparent to the users of the network. As described in the following, the present invention provides a method and apparatus that meet these criteria.

Drawing Description Text (19):
FIG. 15 is a flow chart illustrating the logic used by the naming service manager to process mapping information indicating that a new IP address has been assigned to a computer connected to the LAN shown in FIG. 1;

Detailed Description Text (7):
The network server 30 also includes a processing unit 38, a display 40 and a mass memory 42. The mass memory 42 generally comprises a random access memory (RAM), read only memory (ROM), and a permanent mass storage device, such as a hard disk drive, tape drive, optical drive, floppy disk drive, or combination thereof. The mass memory 42 stores the program code and data necessary for collecting, maintaining and serving mapping information in accordance with the present invention. More specifically, the mass memory 42 stores the naming service manager 46 which collects, maintains and serves mapping information for all of the computers connected to the LAN 24 and their users. The naming service manager 46 collects user mapping information, i.e., user login names, domain names, computer names and IP addresses, from a plurality of naming service agents 50 and correlates the mapping information into a current computer-to-user assignment or "mapping" for each user of the LAN 24. The naming service manager 46 then provides or serves the correlated mapping information to naming service applications 48 which require such information.

Detailed Description Text (9):
Mass memory 62 of the domain controller server 32 stores either the domain controller agent 64 or the host agent 66 that can be used in conjunction with the naming service manager 46 to maintain updated and accurate mapping information for each user and computer of the LAN 24 at any given time. As will be described in

more detail below, the domain controller agent 64 gathers dynamic user login and logout information. The host agent 66, on the other hand, gathers current IP address assignments for the computers connected to the LAN 24. Once gathered, the domain controller agent 64 or host agent 66, whichever the case may be, transmits the mapping information to the naming service manager 46 for further processing. Although both the domain controller agent 64 and the host agent 66 are shown in FIG. 3A, it will be appreciated that only one or the other is normally employed by the naming service manager 46. For example, if dynamic user-to-computer mapping and computer-to-IP address mapping are desired, the domain controller agent 64 is employed. However, if user-to-computer assignments are to remain static, but dynamic computer-to-IP address mapping is still desired, the host agent 66 is employed. Although the host agent 77 is described herein as being located on the domain controller server 60, those of ordinary skill in the art will recognize that the host agent may be located on any suitable computer connected to the LAN 44. It will also be appreciated that many other types of agents may be employed by the present invention, and that the domain controller agent 64 and the host agent 66 are merely illustrative examples of such naming service agents 50.

Detailed Description Text (10):
The domain controller agent 64 and the host agent 66 are referred to as "dynamic sources" of mapping information because they gather mapping information, i.e., computer-to-user and computer-to-IP address assignments or mappings, that are dynamically updatable as users log into and out of computers connected to the LAN 24 and as IP addresses for computers connected to the LAN 24 change. In contrast, the present invention also implements naming service agents 50 that are referred to as "static sources" of mapping information because they provide static or permanent computer-to-user and computer-to-IP address assignments that do not change as users log into and out of computers connected to the LAN 24. As will be described below, the only way in which a static computer-to-user or computer-to-IP address assignment can be changed is if a static naming service agent (rather than a dynamic naming service agent) provides the naming service manager 46 with a new assignment. Those of ordinary skill in the art will appreciate that virtually any application program interface that allows a network administrator to assign network users to network computers and/or to assign network computer names to IP addresses is a static source of mapping information and thus, can serve as a naming service agent 50 to the naming service manager 46. An example of such a static source of mapping information is disclosed in commonly assigned U.S. Patent Application entitled METHOD AND APPARATUS FOR MANAGING INTERNETWORK AND INTRANETWORK ACTIVITY, filed Apr. 2, 1997, to Abraham et al. (the "Abraham application"), the disclosure and drawings of which are specifically incorporated herein by reference.

Detailed Description Text (16):
The mapping information gathered by the naming service agents 50 and provided to the naming service manager 46 is maintained by the naming service manager 46 in a host mapping table 52. The host mapping table 52 is shown in more detail and in FIG. 5A. The host mapping table 52 consists of a plurality of records containing mapping information for each computer connected to the LAN 24. More specifically, each record includes a field for storing the computer name, the IP address assigned to that computer name, the login name of the user currently utilizing the computer and the domain name for the computer. It will be appreciated that the domain name stored in the record may not necessarily be the fully qualified domain name of the computer. In the actual embodiment of the present invention described herein, the domain name is only the name of the domain to which the LAN 24 belongs. The record also includes a logged in flag, which when set, indicates that the user identified in the record by login name is logged in to the computer identified in the record. In addition, a static source flag is provided, which when set, indicates that the mapping information contained in the record was provided to the naming service manager 46 by a static source for such information, i.e., a naming service agent 50 that provides permanent or static user-to-computer assignments to the naming service manager 46. If not set, the static source flag indicates that the mapping

information contained in the record was provided by a dynamic source, i.e., a naming service agent 50 that provides dynamic or changeable naming service information. Finally, each record contains an in-use flag, which when set, indicates that the record is an active record, and thus may be served to naming service applications 48.

Detailed Description Text (25):
FIG. 8 illustrates the logic used by the initial state generator for the domain controller agent 64. The logic begins in a block 170 and proceeds to a block 172 where the domain controller agent acquires an initial list of computers in active session with the LAN 24 and into which users have logged, from the domain controller server 32. In a block 174, the domain controller agent 64 performs a NETBIOS query to acquire the IP address for each computer in the initial list. Those of ordinary skill in the art will recognize that NETBIOS is an application program interface used to provide other application programs with computer-to-IP address assignments and with a uniform set of commands for requesting lower level network services required to conduct sessions between computers connected to the LAN 24, so that the computers may transmit data back and forth via the LAN 24.

Detailed Description Text (29):
In this regard, the domain controller agent 64 obtains the first computer identified in the combined list in a block 194. In a block 196, the domain controller agent 64 generates a transaction record 53 containing the domain name, computer name, and IP address of the computer as well as the login name of the user assigned to the computer. In a decision block 198, the domain controller agent 64 determines if computer is a newly active computer. If so, the domain controller agent 64 identifies the transaction record 53 as a login update record and stores the login update record in output queue in a block 200. Otherwise, the domain controller agent 64 identifies the transaction record 53 as a logout update record and then stores the logout update record in output queue in a block 202. Ultimately, the logic proceeds to a decision block 204 where the domain controller agent 64 determines if the last computer in the combined list has been processed. If not, the next computer in the combined list is obtained in a block 205 and blocks 196-206 are repeated for each computer in the combined list so that either a login transaction record or a logout transaction record is stored in the transaction container 54, and hence in the output queue of the domain controller agent 64.

Detailed Description Text (36):
Returning to decisions block 238, if the computer being processed is not newly active or newly inactive, the logic proceeds to a decision block 242 where the host agent 66 determines if a new IP address for the computer has been assigned. If so, the host agent 66 generates and stores two different transaction records 53 in the output queue in a block 244. The first transaction record 53 is identified as a prior address update and contains the former IP address for the computer and its computer name. The second transaction record 53 is identified as a current address update and contains the new IP address of the computer and its computer name.

Detailed Description Text (45):
Returning to decision block 116, if the received transaction container 54 does not contain an agent registration request, an application registration request, an agent unregistration request, or an application unregistration request, the logic proceeds to a decision block 120 where it determines if the transaction container 54 contains a query from a naming service agent 50 or a naming service application 48 for mapping information. If so, the logic proceeds to a decision block 122 where it determines if the host mapping table 52 maintained by the naming service manager 46 contains a record having the mapping information requested. For example, if the naming service application 48 or agent 50 is seeking mapping information for a particular user, i.e., the computer name, domain name and IP address assigned to a particular user, the naming service manager 46 determines in a decision block 122

if the host mapping table 52 includes a record having the same login name as the login name provided by the querying application or agent in the header 55 of the transaction container 54. If so, the naming service manager 46 returns the record to the requesting agent 50 or application 48 in a block 124. Similarly, if the naming service application 48 or agent 50 is seeking mapping information for a particular computer, the naming service manager 46 determines in decision block 122 if the host mapping table 52 includes a record having the same computer address or IP address provided by the querying application or agent in the header 55 of the transaction container 54. If so, the naming service manager 46 returns the record to the requesting agent 50 or application 48 in block 124. However, if no such record is found in the host mapping table 52 in either case, the naming service manager 46 returns an invalid record to the requesting agent or application in a block 126. The logic then returns to decision block 102 and the naming service manager 46 waits for another transaction container 54

Detailed Description Text (49):
It will be appreciated that if a prior address update transaction record is received by the naming service manager 46, the naming service manager 46 must update the host mapping table 52 to reflect that the IP address for a computer connected to the LAN 24 has gone out of scope, i.e., that for one reason or another, the computer is no longer associated with the IP address found in the prior address update record. However, if the host mapping table 52 indicates that a user is logged into the computer assigned to the prior IP address, the user must be logged out of the computer at the prior IP address before the mapped record in the host mapping table 52 can be updated. Accordingly, if the result of decision block 284 is positive, the naming service manager 46 clears the logged in flag and the static source flag in the located record of the host mapping table in a block 286. In a block 288 the naming service manager generates and stores a transaction record in the naming service manager output queue that is identified as a logout update record. The logout update record includes the computer name, domain name and IP address of the computer identified in the host mapping record as well as the login name of the user.

Detailed Description Text (53):
It will be appreciated that when a current address update record is received by the naming service manager 46, the host mapping table 52 must be updated to reflect that a new IP address has been assigned to a computer connected to the LAN 24. Accordingly, the current address update record contains the computer name of the computer and the new or current IP address that has been assigned to the computer. The other fields in the current address update record may contain data, but that data is not necessarily valid.

Detailed Description Text (55):
If a host mapping record in the host mapping table 52 containing the current computer name is located, or if no such record is found, but a record has been added, the logic proceeds to a decision block 302. In decision block 302, the naming service manager 46 determines if the host mapping record contains a valid IP address different than the current IP address specified in the current address update record. In other words, the naming service manager 46 determines if the computer specified in the host mapping record had a different prior IP address. If so, that assignment must be removed so that the current IP address may be assigned to the computer identified in the current address update record. Consequently, the other record in the host mapping table 52 having the current IP address specified in the current address update record is processed as a prior address update record in a block 304. As a result of processing a record as a prior address update record in accordance with the logic shown in FIG. 14, the IP address found in the host mapping record will be invalidated, thus making the current IP address freely available.

Detailed Description Text (57):

Once the host mapping table 52 has been updated with the current IP address as described above, the naming service manager 46 determines in a block 310 if the IP address in the retrieved host mapping record was an invalid address prior to the update. If not, no further processing is necessary and the logic ends in a block 318. However, if the prior IP address was invalid, the change to the current IP address may signify that a user has logged in, but generation of a login update record has been deferred until a valid IP address has been assigned. In this regard, the logic proceeds from decision block 310 to a decision block 312 where the naming service manager 46 determines if the logged in flag in the identified host mapping record is set. If not, a user has not logged into the identified computer so a login record is not necessary. Consequently, the logic merely ends in block 318. However, if the logged in flag in the host mapping record is set, a user has logged in and thus, it may be necessary to output a login record. Accordingly, the logic determines in a block 314 if the IP address in the host mapping record is now a valid IP address, i.e., the IP address in the host mapping record has changed from an invalid address to a valid address. If not, the IP address is still invalid, and a login update record in not required. However, if the IP address is now a valid one, the logic proceeds to a block 316 where it generates and stores a transaction record 53 in the naming service manager's output queue. The transaction record is identified as a login record and contains the computer name, IP address, domain name and login name found in the host mapping record. The logic then ends in a block 318.

Detailed Description Text (65):
If a record having the login computer name is found in the host mapping table 52 or if such a record has been added to the host mapping table 52, the logic proceeds to a decision block 354 where the naming service manager 46 determines if the IP address in the login update record is different than the IP address identified in the host mapping record or if the IP address identified in the login record is a new IP address that cannot be found in the host mapping table 52. If either of these conditions is met, then the computer identified by the computer name stored in the host mapping record and the login update record has been assigned a new IP address and the host mapping table 52 must be updated accordingly. However, in order to avoid unnecessary processing, the naming service manager 46 determines if the login IP address is valid in a decision block 356 before further processing the new IP address. Hence, if the login IP address is new or changed and is valid, the login updated record is processed as a current address update record in a block 358. As a result of the logic illustrated in FIG. 15, the host mapping record having the computer name specified in the login update record is updated to the login IP address.

Detailed Description Text (71):
Returning to decision block 378, if the IP address provided in the login update record is valid, the logic proceeds to a block 382. In a block 382, the naming service manager 46 updates the host mapping record with the computer name, IP address, login name and domain name specified in the login update record. In addition, the logged in flag is set and the static source flag is set or cleared to reflect the source of the login update record, i.e., a dynamic source of mapping information or a static source of mapping information. Next, in a decision block 384, the naming service manager 46 determines if the IP address in the host mapping record updated in block 382 is valid, if not, the naming service manager 46 defers generating and outputting a login update record to the naming service applications 48 until a valid IP address is assigned during a current address update as described above. Consequently, the logic ends in a block 388. However, if the IP address in the host mapping record is valid, the naming service manager 46 finally generates a login update record containing the computer name, IP address, domain name and login name from the host mapping record in a block 386. The logic then ends in block 388.

ABRA

Previous Doc Next Doc Go to Doc#
First Hit Fwd Refs

Generate Collection

L3: Entry 1 of 1 File: USPT Nov 2, 1999

DOCUMENT-IDENTIFIER: US 5978568 A
TITLE: Method and apparatus for resolving network users to network computers

Abstract Text (1):
A naming service manager (46) is provided for resolving mapping information regarding a plurality of computers connected to a local area network (LAN 24) and their users. The naming service manager 46 collects mapping information, i.e., user login names, domain names, computer names and IP addresses, from a plurality of naming service agents 50 located on the LAN (24) and correlates the mapping information into a current computer-to-user assignment or "mapping" for each user of the LAN 24 and/or a current IP address-to-computer assignment or "mapping" for each computer connected to the LAN 24. The naming service manager (46) serves the correlated mapping information to a plurality of naming service applications (48) which process the correlated mapping information in accordance with their own needs and requirements.

Current US Original Classification (1):
709/224

Previous Doc Next Doc Go to Doc#

Priority 10/5/99

US005978568A

United States Patent [19]

Abraham et al.

[11] **Patent Number: 5,978,568**

[45] **Date of Patent: Nov. 2, 1999**

[54] **METHOD AND APPARATUS FOR RESOLVING NETWORK USERS TO NETWORK COMPUTERS**

[75] Inventors: **Dalen M. Abraham**, Redmond; **Todd A. Barnes**, Snohomish; **Mark G. Grieve**, Bellevue, all of Wash.

[73] Assignee: **Sequel Technology Corporation**, Bellevue, Wash.

[21] Appl. No.: **08/826,598**

[22] Filed: **Apr. 3, 1997**

Related U.S. Application Data

[60] Provisional application No. 60/040,424, Mar. 11, 1997.

[51] **Int. Cl.**^{6} **G06F 13/00**

[52] **U.S. Cl.** **395/200.54**

[58] **Field of Search** .... 364/DIG. 1 MS File, 364/DIG. 2 MS File; 707/10; 395/200.3, 200.33, 200.47, 200.48, 200.5, 200.51, 200.52, 200.53, 200.54, 200.75

[56] **References Cited**

U.S. PATENT DOCUMENTS

5,347,633	9/1994	Ashfield et al.	395/200.68
5,377,323	12/1994	Vasudevan	395/200.56
5,425,028	6/1995	Britton et al.	395/200.58
5,742,769	4/1998	Lee et al.	395/200.36
5,813,006	9/1998	Polnerow et al.	707/10

OTHER PUBLICATIONS

"Traffic Reduction of Name Server Lookups over Low-Bandwidth Connections,"IBM Technical Disclosure Bulletin, Vol. 39, No. 12, Dec. 1996, pp. 221–222.

W. Doeringer, D. Dykeman, M. Peters, H. Sandick, K. Vu, J. Derby, "Access Architecture for a Multiprotocol Broadband Backbone,"*Computer Networks and ISDN Systems*, Vol. 29, No. 2, Jan. 1997, pp. 137–155.

Primary Examiner—Robert B. Harrell
Attorney, Agent, or Firm—Christensen O'Connor Johnson & Kindness PLLC

[57] **ABSTRACT**

A naming service manager (46) is provided for resolving mapping information regarding a plurality of computers connected to a local area network (LAN 24) and their users. The naming service manager **46** collects mapping information, i.e., user login names, domain names, computer names and IP addresses, from a plurality of naming service agents **50** located on the LAN (24) and correlates the mapping information into a current computer-to-user assignment or "mapping" for each user of the LAN **24** and/or a current IP address-to-computer assignment or "mapping" for each computer connected to the LAN **24**. The naming service manager (46) serves the correlated mapping information to a plurality of naming service applications (48) which process the correlated mapping information in accordance with their own needs and requirements.

52 Claims, 20 Drawing Sheets